UNIVERSIDADE FEDERAL DE SÃO CARLOS
DEPARTAMENTO DE COMPUTAÇÃO
CIÊNCIA DA COMPUTAÇÃO

Júlia Aparecida Sousa de Oliveira

Detecção de Discurso de Ódio: Análise de Modelos Clássicos e Redes Neurais com Estratégias de Balanceamento

São Carlos - SP

2025

Júlia Aparecida Sousa de Oliveira

Detecção de Discurso de Ódio: Análise de Modelos Clássicos e Redes Neurais com Estratégias de Balanceamento

Trabalho de Conclusão de Curso apresentado ao curso de Ciência da Computação da Universidade Federal de São Carlos, como requisito para a obtenção do título de Bacharel em Ciência da Computação.

Orientação Prof. Dr. Alan Demétrius Baria Valejo

São Carlos - SP

2025

Dedico este trabalho à minha mãe, minha maior inspiração

Agradecimentos

Nesse espaço, gostaria de expressar minha gratidão por muitas das pessoas que conheci e pelos grupos dos quais fiz parte. Essas experiências moldaram minha formação pessoal e profissional.

Inicialmente, agradeço à minha mãe, por ser meu maior exemplo e incentivadora em todas as iniciativas que eu me propuser. Se tive a possibilidade de seguir na escolha de um caminho, foi pelo sacrifício dela, que sempre foi meu maior suporte.

Também tenho imenso reconhecimento pelo grupo de amigos que escolhi como família durante esta trajetória. Com vocês, aprendo continuamente sobre acolhimento, compreensão e parceria nas diversas dificuldades. Deixo um agradecimento especial a Igor, Bárbara, Rafael e Jayme, por estarem comigo em todos os momentos.

Agradeço igualmente ao meu orientador, Prof. Dr. Alan Valejo, pela oportunidade de participar de Projeto Integrador-Extensionista juntamente com iniciativa privada, tutoria em fundação e gestão do Panda, pelo espaço no laboratório do Midas e, por fim, pela orientação no presente Trabalho de Conclusão de Curso.

Aos membros dos grupos de extensão que participei, como a CATI Jr, a Semana Acadêmica de Computação (SECOMP), as Pyladies e o Panda — onde tive muitos momentos de aprendizado técnico, vivências, trocas e pude experimentar o verdadeiro sentido de pertencimento.

Por último, deixo aqui uma aspiração à minha *Sangha* do Centro de Estudos Budistas (CEBB) e, em especial, à minha tutora Rosana, por serem guia e amparo na impermanência da vida.

"Por um mundo onde sejamos socialmente iguais,
humanamente diferentes e totalmente livres."

(Rosa Luxemburgo)

Resumo

Com o crescimento do acesso à internet e consequentemente às redes sociais, surgem novas oportunidades de comunicação, interação e troca de informações numa escala à nível global. Porém, esse ambiente virtual também possibilitou a disseminação de conteúdos negativos como o discurso de ódio. Ele pode ser compreendido como qualquer manifestação que promova a violência e discriminação contra indivíduos ou grupos com base em características como raça, religião, gênero, orientação sexual, etc. Nesse sentido, o presente trabalho teve como foco uma avaliação de modelos de aprendizado de máquina, comparando algoritmos tradicionais e redes neurais aplicados à dados textuais extraídos de redes sociais contendo discurso de ódio. Fez-se uso de técnicas de balanceamento, como *class wieght*, *oversampling* e *undersampling* como maneira de lidar com a diferença da quantidade de observações entre classes. Os resultados demonstraram que os classificadores tradicionais apresentaram maior consistência nas métricas de F-Score ponderado e AUC, mesmo no cenário original sem tratamento de balanceamento, chegando a valores superiores a 0.93. Enquanto que os modelos baseados em redes neurais, como MLP, CNN e LSTM, se mostraram mais sensíveis ao tipo de balanceamento, sugerindo necessidades de ajustes para melhoria da capacidade de generalização.

Palavras-chave: discurso de ódio, aprendizado de máquina, balanceamento de classes.

Abstract

With the growth of internet access and, consequently, the expansion of social media, new opportunities for communication, interaction, and information exchange have emerged on a global scale. However, this virtual environment has also enabled the spread of negative content, such as hate speech. Hate speech can be understood as any expression that promotes violence or discrimination against individuals or groups based on characteristics such as race, religion, gender, sexual orientation, among others. In this context, the present study focused on evaluating machine learning models by comparing traditional algorithms and neural networks applied to textual data extracted from social media containing hate speech. Balancing techniques such as class weight, oversampling, and undersampling were employed as a way to handle class imbalance. The results showed that traditional classifiers demonstrated greater consistency in the weighted F-Score and AUC metrics, even in the original scenario without balancing treatment, reaching values above 0.93. On the other hand, neural network-based models, such as MLP, CNN, and LSTM, proved to be more sensitive to the type of balancing, suggesting the need for adjustments to improve their generalization capacity.

Keywords: hate speech, machine learning, class balancing.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AUC	Area Under the Curve
Bi-LSTM	Bi-Directional Long Short Term Memory
CNN	Rede Neural Convolucional
DL	Aprendizado Profundo
EFB	Exclusive Feature Bundling
GBM	Gradient Boosting Machines
GOSS	Gradient-based One-Side Sampling
LSTM	Long Short-Term Memory Network
ML	Aprendizado de Máquina
MLP	Perceptron Multicamadas
OvR	One-vs-Rest
PLN	Processamento de Linguagem Natural
RNN	Redes Neurais Recorrentes
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine
TF-IDF	Term Frequency-Inverse Document Frequency
URL	Uniform Resource Locator

Sumário

1 Introdução

No contexto de globalização e acesso à informação, teve-se o ganho de espaço das redes sociais como formas de comunicação, expressão de ideias e formação de comunidades. Essa facilidade gerou benefícios como conexões com pessoas, compartilhamento e trocas, conteúdo de entretenimento e incentivo econômico de divulgação e comercialização de bens e serviços.

Entretanto, observa-se também um fenômeno crescente da propagação das chamadas *fake news* e de discurso de ódio contra grupos minoritários na sociedade por meio de postagens, compartilhamentos e comentários com conotações negativas. Dados recentes do relatório Global de 2023 da Statista (2023) apontaram que cerca de 41% dos usuários da internet já relataram ter visto ou sofrido discurso de ódio online.

Em vista disso, torna-se inviável a detecção e análise manual de grandes volumes de conteúdo, considerando que são gerados continuamente. Em um estudo feito por Silva e Roman (2020), aponta-se as redes sociais do Facebook e Twitter como as principais plataformas usadas atualmente, e que estudos de identificação de discurso de ódio se apoiam em conteúdos extraídos das mesmas.

Então, técnicas automatizadas baseadas em algoritmos de Aprendizado de Máquina (ML) vem sendo utilizadas para a identificação de padrões e classificação de conteúdos como discurso de ódio ou não, de acordo com Schmidt e Wiegand (2017). Também observa-se um crescimento do uso de modelos de Aprendizado Profundo para tarefas de processamento de linguagem natural. Uma necessidade apontada é a necessidade de comparação e avaliação do uso dessas técnicas em cenários de aplicação de balanceamento de dados, para maior compreensão do comportamento dos modelos em diferentes contextos e conjunto de dados, de acordo com MacAvaney et al. (2019).

1.1 Objetivo

Tendo essa contextualização, a detecção automática de discurso de ódio se torna uma tarefa com relevância e importância social na promoção de espaços seguros nas redes sociais e demais plataformas.

Nesse sentido, o presente trabalho tem como objetivo geral fazer uma avaliação e comparação do desempenho de algoritmos de classificação, incluindo modelos tradicionais de Aprendizado de Máquina (ML) e arquiteturas de Aprendizado Profundo (DL) aplicados à dados textuais extraídos de redes sociais para a tarefa de detecção de discurso de ódio. Além disso, busca-se observar o comportamento de tais algoritmos em relação as formas

de representação vetorial de texto e técnicas de balanceamento como fatores na qualidade dos modelos de predição.

1.2 Objetivo específico

Como objetivos específicos, tem-se a tratativa dos dados em uma etapa de pré-processamento e limpeza do corpus para garantia da qualidade das observações em análise posterior. Em sequência, as transformações para representações com uso de técnicas de *Term Frequency-Inverse Document Frequency* (TF-IDF) e *embeddings*, de modo a obter as entradas ideiais de acordo com o modelo utilizado e sua construção.

A etapa de treinamento será feita com a implementação dos algoritmos de Aprendizado de Máquina, como Naive Bayes, Regressão Logística, SVM, Floresta Aleatória, LightGBM e MLP, todos utilizando vetores derivados de TF-IDF para ter a classificação da classe minoritária contendo discurso de ódio.Para a implementação das redes neurais profundas, os dados devem passar por um processo de tokenização, criação de vocabulário e preenchimento para modelos como Rede Neural Convolucional (CNN) com uso de *embeddings*.

Na etapa posterior, serão utilizadas estratégias de balanceamento de classes como *oversampling*, *undersampling* e a utilização de *class weight*, a fim de lidar com os efeitos de distribuições dos dados nas bases em análise. A avaliação será feita com as métricas de F1-Score ponderado, valor de AUC e acurácia, como comparação de desempenho obtido pelos modelos nos experimentos.

2 Fundamentação Teórica

Este capítulo contém os conceitos que são abordados neste trabalho. Além disso, serve como um guia para entender os assuntos abordados.

2.1 Mineração de Dados

Considerando o cenário atual com avanços contínuos da tecnologia da informação, observa-se o crescimento da geração, armazenamento e disponibilidade de dados tanto em organizações e instituições quanto online, como em redes sociais. Ao lidar com essa grande quantidade de dados, existe o processo de manipulação e análise chamado de *mineração de dados*. Esse passo é definido por Kaufman e Rousseeuw (2009) como "(...) é definido como o processo de descobrimento de padrões nos dados. O processo deve ser automático ou (mais comumente) semiautomático. Os padrões descobertos devem ser significantes de modo que levem a alguma vantagem, geralmente uma vantagem econômica".

Assim, entende-se que o uso de *mineração de dados* objetiva ter resultados dessa transformação dos dados em informações consideradas úteis para solução de problemas e tomadas de decisões estratégicas.

Tendo em vista que parte desse grande volume de dados se encontra em formato textual, tem-se que o processo de *mineração de texto* se torna uma tarefa essencial, especializada pela diferente natureza dos dados. Essa tarefa lida com dados não estruturados, uma vez que se trata de textos escritos em língua natural, envolvendo atividades de pré-processamento para transformação e representação — devendo levar em consideração o aspecto semântico para extração de conhecimento em análises posteriores (SINOARA; MARCACINI; REZENDE, 2021).

2.2 Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN) é considerado uma área multidisciplinar, envolvendo aspectos de aplicação de áreas diversas como a Computação, Linguística, Estatística, Psicologia, entre outras. É um campo de estudo amplo sobre o desenvolvimento de algoritmos que possam compreender e interpretar a linguagem natural humana. Sua complexidade se estende pela variedade de idiomas, dialetos, gírias e particularidades de estrutura gramatical, sintaxe e vocabulário que cada língua possui em sua composição.

Quando pensamos em linguagem, é importante lembrar do contexto em que ela está inserida. Afinal, os textos podem ter elementos como sarcasmo, metáforas e ambiguidades,

que são comuns na comunicação entre as pessoas, mas que podem não ser entendidos por uma máquina.

2.3 Representação de Dados Textuais

2.3.1 Pré-processamento

Para a realização de tarefas de Processamento de Linguagem Natural com conteúdos textuais derivados de postagens em redes sociais como *tweets*, deve ser um atenção aumentada ao pré-processamento textual para garantia de qualidade das representações que devem ser usadas como entradas paras os modelos. De acordo com Gimpel et al. (2011), o texto que foi extraído de uma rede social apresenta aspectos de informalidade, como ruído, uso de gírias, *hashtags*, abreviações, *emojis* ou caracteres especiais e marcações. Sendo assim, a primeira etapa seria uma remoção desses "excessos"para ter o dado textual que seja relevante de maneira semântica.

Após essa limpeza, tem-se o passo de tokenização, que segmenta o texto em unidades menores (os chamados tokens) que podem ser de palavras ou símbolos. Em seguida, pode-se utilizar técnicas de normalização como lematização ou *stemming* que quando aplicadas reduzem as palavras à forma canônica, para reduzir a dimensionalidade do vocabulário e melhorar a capacidade de generalização do classificador. Outro passo a ser considerado é a remoção de *stopwords*, palavras comuns como preposições, artigos e pronomes que não agregam semanticamente ao texto e podem não ter relevância para a tarefa de detecção.

2.3.2 Tokenização e Vetorização

Como já mencionado anteriormente, os dados textuais precisam passar por transformações para serem usados como entrada em modelos de aprendizado de máquina e aprendizado profundo que exigem o formato de dados numéricos. Isso ocorre porque utilizam operações matemáticas como multiplicações, somas, gradientes e vetores para tarefas de classificação e predição Jurafsky e Martin (2023).

Uma das possibilidades de transformação dos dados é o uso da técnica *Term Frequency-Inverse Document Frequency* (*TF-IDF*). Essa métrica utiliza dois conceitos principais: i) cálculo da frequência do termo (*Term Frequency*), uma medida da quantidade de aparições de um termo em um documento em relação ao total de palavras presentes no mesmo; ii) cálculo da frequência inversa do documento (*Inverse Document Frequency*), que consiste em uma forma de penalização para termos muito frequentes em vários documentos Manning, Raghavan e Schütze (2008).

A fórmula geral combinando esses dois conceitos é apresentada abaixo:

$$TF\text{-}IDF(t, d, D) = TF(t, d) \times IDF(t, D) \tag{2.1}$$

em que:

- t = termo,
- d = documento,
- D = coleção de documentos.

Outra forma de transformação são os chamados *word embeddings*, que são representações densas e vetoriais em um espaço contínuo de baixa dimensão. Essas representações são capazes de capturar características semânticas, aproximando palavras com significados semelhantes, uma abordagem distinta do uso de representações esparsas como o *TF-IDF*.

A geração de *embeddings* resulta em representações aprendidas a partir de grandes corpora, permitindo que relações semânticas e sintáticas — como sinônimos, analogias e padrões gramaticais — sejam extraídas dos textos. Um exemplo clássico para demonstrar a propriedade de analogias vetoriais é mostrado a seguir:

$$king - man + woman \approx queen \tag{2.2}$$

Essa relação ocorre porque os *embeddings* são treinados para prever contextos (*Skip-Gram*, *CBOW*) ou coocorrências globais (*GloVe*), estruturando um espaço vetorial que aproxima semanticamente palavras semelhantes Pennington, Socher e Manning (2014).

Os modelos Word2Vec, por exemplo, utilizam duas arquiteturas principais para esse fim. O modelo *Continuous Bag-of-Words* (CBOW) aprende a predição de uma palavra-alvo a partir de seu contexto. Por exemplo, na frase “o gato sentou no tapete”, o CBOW utilizaria as palavras de contexto {“o”, “gato”, “no”, “tapete”} para prever a palavra central “sentou”. Em contrapartida, o modelo *Skip-Gram* realiza a tarefa inversa: a partir de uma palavra de entrada, ele tenta prever as palavras de seu contexto. Revendo sobre o mesmo exemplo anterior, ao receber a palavra “sentou”, o Skip-Gram aprenderia a prever palavras como {“o”, “gato”, “no”, “tapete”}. De maneira geral, o Skip-Gram apresenta um desempenho melhor para palavras raras e em bancos de dados menores, enquanto o CBOW tende a ser mais rápido e ligeiramente melhor para palavras frequentes Mikolov et al. (2013).

2.4 Algoritmos de Aprendizado de Máquina

Nessa seção serão apresentados os principais conceitos do algoritmos de aprendizado de máquina utilizados, descrevendo suas características, uso do dados e saídas esperadas na classificação desses.

2.4.1 Modelos Lineares

A regressão logística é um modelo linear utilizado para problemas de classificação que envolvam uma variável dependente do tipo categórica, podendo ser binária ou multiclasse. Nesse método, modela-se a probabilidade de ocorrência de um evento por meio da função sigmoide, restringindo a saída ao intervalo [0,1] (HOSMER; LEMESHOW; STURDIVANT, 2013).

Para a tarefa de classificação de texto, a regressão logística apresenta bom desempenho em vetores esparsos de alta dimensionalidade, como os derivados de *TF-IDF* ou *word embeddings*, de acordo com Jurafsky e Martin (2023). No caso de classificação multiclasse, pode-se utilizar a estratégia de *One-vs-Rest (OvR)* ou *Softmax Regression*, que generaliza a função para múltiplas categorias.

Outro algoritmo supervisionado amplamente empregado é o *Support Vector Machine (SVM)*, que tem como objetivo encontrar o hiperplano ótimo que maximiza a margem entre as classes em um espaço de características. Quando se utiliza o *linear kernel*, assume-se que os dados são linearmente separáveis, operando diretamente sobre os vetores originais sem necessidade de projeções para aumento de dimensionalidade (CORTES; VAPNIK, 1995).

Fundamentalmente, o SVM identifica os *support vectors*, isto é, os exemplos mais próximos do hiperplano que definem a fronteira de decisão. Essa abordagem confere robustez em cenários de alta dimensionalidade e baixa densidade de dados relevantes, como ocorre em tarefas de classificação de texto com representações em *TF-IDF*. Vale ressaltar que o SVM pode ser adaptado para problemas multiclasse por meio de extensões como *One-vs-One* ou *One-vs-Rest* (JOACHIMS, 1998).

2.4.2 Modelos Baseados em Árvores e Ensemble

Os modelos baseados em árvores são amplamente utilizados em tarefas de classificação pela capacidade de lidar com dados não linearmente separáveis e capturar interações de maior complexidade entre as variáveis.

A Floresta Aleatória é um algoritmo do tipo *ensemble* que utiliza múltiplas *decision trees* para reduzir a variância do modelo individual e favorecer a capacidade de generalização (BREIMAN, 2001). Cada árvore é treinada em uma amostra selecionada aleatoriamente

do conjunto de dados (processo conhecido como *bootstrap*) e, na construção de cada nó, um subconjunto aleatório de variáveis é considerado, gerando árvores decorrelacionadas. A previsão final é obtida por meio de uma votação majoritária no caso de classificação.

Da otimização de *Gradient Boosting Machines (GBMs)* tem-se a implementação do *LightGBM*, no qual a construção dos modelos ocorre de forma sequencial, sendo que cada árvore busca corrigir os erros residuais do conjunto anterior (FRIEDMAN, 2001).

Além disso, o *LightGBM* faz uso de técnicas como *Gradient-based One-Side Sampling (GOSS)* e *Exclusive Feature Bundling (EFB)*, que visam melhorar a eficiência computacional e a escalabilidade em conjuntos de dados de alta dimensão (KE et al., 2017). Dessa forma, o *LightGBM* torna-se adequado para tarefas de classificação de texto com grandes volumes de dados e vetores esparsos.

2.4.3 Modelos Probabilísticos

Dentro dos vários modelos probabilísticas, destaca-se o uso do Naive Bayes Multinomial em estudos de classificação de texto. Tem-se que a principal característica dessa abordagem é assumir uma hipótese de independência condicional entre as variáveis preditoras, em outras palavras, estabelece-se que cada característica contribui de maneira independente para a probabilidade de determinada classe (MANNING; RAGHAVAN; SCHüTZE, 2008).

Além disso, o modelo incorpora explicitamente a **probabilidade a priori** (*prior probability*) das classes no cálculo da probabilidade posterior, conforme o Teorema de Bayes, dado por:

$$P(C_k|X) = \frac{P(X|C_k) \cdot P(C_k)}{P(X)} \tag{2.3}$$

em que $P(C_k|X)$ é a probabilidade posterior da classe C_k dado o vetor de atributos X; $P(X|C_k)$ representa a verossimilhança dos atributos para a classe C_k; $P(C_k)$ é a probabilidade a priori da classe; e $P(X)$ é a probabilidade total dos atributos. Assim, a predição considera tanto a frequência de ocorrência de termos quanto a distribuição pré-existente das classes, o que torna o modelo robusto mesmo em cenários com dados desbalanceados (SEBASTIANI, 2002).

2.5 Arquiteturas de Aprendizado Profundo

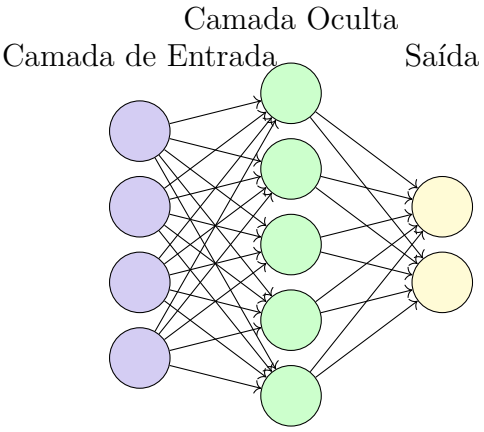
Nessa seção serão trazidos os conceitos de arquitetura aplicadas ao Processamento de Linguagem Natural, descrevendo sua construção, comportamento esperado e saída da classificação.

2.5.1 Perceptron Multicamadas (MLP)

O Perceptron Multicamadas (*MLP*) é uma arquitetura de rede neural *feedforward* que é composta por uma ou mais camadas ocultas, sendo que cada neurônio aplica uma transformação linear acompanhada em seguida de uma de ativação não linear. Assim, a estrutura possibilita que o modelo aprenda padrões não lineares entre as variáveis de entrada e de saída.

Para a tarefa de classificação de texto, o modelo deve ser aplicado sobre representações densas como *word embeddings* agregados ou sobre vetores de características resultantes da transformação com a técnica de *TF-IDF*. Ele não possui a capacidade de capturar relações sequenciais entre palavras, mas pode ser eficaz como classificador, desde que combinado com camadas de regularização e funções de ativação como *ReLU*, segundo Goodfellow, Bengio e Courville (2016). A Figura 1 ilustra a arquitetura de um Perceptron Multicamadas.

Figura 1 – Arquitetura de um *Perceptron Multicamadas* (MLP) para classificação de texto



Fonte: Adaptado de Goodfellow, Bengio e Courville (2016).

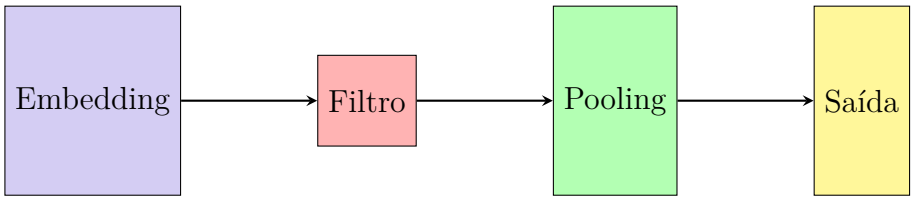
2.5.2 Redes Neurais Convolucionais (CNNs)

No desenvolvimento das Redes Neurais Convolucionais (CNNs) tem sua origem como objetivo para processamento de dados com estrutura espacial local, como imagens de acordo LeCun et al. (1998). Para o caso de classificação de texto, o uso das CNNs se faz com a aplicação de filtros convolucionais (Conv1D) para conseguir captar as características locais de maneira sequencial das palavras. Esse processo seria o equivalente a extrair n-grams que fossem considerados relevantes de maneira automática.

Na arquitetura para dados textuais inclui-se uma camada de embedding, seguida por camadas Conv1D e uma camada de preenchimento (exemplo do GlobalMaxPooling 1D) retendo as características mais significativas da sequência. Tendo essa combinação,

o modelo se torna menos sensível à posição exata das palavras na sentença em análise (GOLDBERG, 2016). A Figura 2 apresenta essa arquitetura de uma CNN aplicada à classificação de texto.

Figura 2 – Arquitetura de uma *Rede Neural Convolucional* (CNN) aplicada à classificação de texto.



Arquitetura CNN com *Conv1D* e *Pooling*

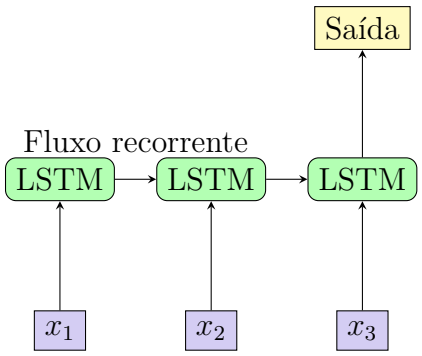
Fonte: Adaptado de Kim (2014)

2.5.3 Redes Neurais Recorrentes (LSTM)

Por fim, outra abordagem é o uso de Redes Neurais Recorrentes (RNNs) que possuem em seu objetivo lidar com dados sequenciais, processando entradas de modo parcial enquanto mantêm um estado que carrega informações de contexto. Porém, as RNNs tradicionais enfrentam o problema de dependência de longo prazo devido ao comportamento do gradiente durante a etapa de processamento. Por isso, surge a Long Short-Term Memory networks (LSTMs) em que a arquitetura introduz o uso de células de memória e portas de entrada, saída e esquecimento, apropriadas para retenção ou descarte de informações com o passar do tempo (HOCHREITER; SCHMIDHUBER, 1997). A Figura 3 ilustra a estrutura de uma LSTM para processamento de sequência textual.

A Figura 3 ilustra o processamento de uma sequência textual, onde cada entrada x_t é o vetor de uma palavra. Cada célula **LSTM** (verde) processa uma palavra da sequência e atualiza seu estado de memória, que é passado para o passo seguinte através do **fluxo recorrente** (setas horizontais). Esse fluxo permite que a rede retenha informações de longo prazo. Ao final, o estado da última célula é usado pela camada de **Saída** (amarelo) para gerar o resultado final, como uma classificação. Os elementos-chave são a entrada sequencial (x_t), a célula de memória que processa a informação, e o estado recorrente que conecta os passos de tempo.

Figura 3 – Arquitetura de uma *Long Short-Term Memory Network* (LSTM) para processamento de sequência textual.



Fonte: Adaptado de Hochreiter e Schmidhuber (1997)

2.5.4 Treinamento: Minimização de Função de Erro, Backpropagation e Descida do Gradiente

Durante o treinamento de uma rede neural, os pesos das conexões são ajustados com o objetivo de minimizar uma função de erro, sendo que essa é definida como a diferença entre as saídas previstas pelo modelo e os valores reais (*ground truth*).

Essa minimização é realizada por meio do algoritmo de retropropagação do erro(em inglês *back-propagation*), que calcula o gradiente da função de custo em relação a cada peso da rede, propagando os erros de forma reversa, das camadas de saída para as camadas de entrada.

A partir desses gradientes, aplica-se a técnica de **descida do gradiente** (*gradient descent*), que ajusta iterativamente os pesos na direção que reduz o erro. Essa combinação de retropropagação e descida do gradiente é fundamental para a aprendizagem eficaz de padrões complexos em redes neurais profundas (RUMELHART; HINTON; WILLIAMS, 1986; GOODFELLOW; BENGIO; COURVILLE, 2016).

2.6 Técnicas de Balanceamento

Para lidar com o desbalanceamento nas bases de dados, um desafio comum ao lidar com tarefas de detecção de discurso de ódio, o experimento mostrará o desempenho com o uso de três técnicas: *class weight*, *oversampling* e *undersampling*.

O ajuste com *class weight* é basicamente a atribuição de pesos às classes durante o treinamento do modelo, para que assim os erros das classes minoritárias não tenham tanto efeito na função de perda. Nesse caso, a alteração não modifica o número de observações na amostra, mas consegue obter uma redução do viés do modelo predizer melhor a classe

majoritária He e Garcia (2009).

Outra técnica de reamostragem a ser utilizada é a *Synthetic Minority Oversampling Technique (SMOTE)*, que em seu funcionamento gera exemplos sintéticos da classe minoritária fazendo uso de interpolação de novas instâncias entre as amostras reais que se localizam próximas no espaço de atributos. Assim visa-se a diminuição do *overfitting* e melhoria na capacidade de generalização do modelo de classificação.

Além disso, tem-se o *Random Undersampling* que remove de maneira aleatória exemplos da classe majoritária até que atinja um balanceamento em relação à classe minoritária. Assim, o conjunto de dados é reduzido, sendo favorável à redução no tempo de treinamento, porém pode-se descartar informações relevantes Drummond e Holte (2003).

2.7 Métricas de avaliação de modelos

Para a mensuração de desempenho dos modelos devem ser usadas métricas de avaliação, que são indicadores da qualidade ou de performance do experimento em formato numérico. Essas medidas são importantes para detecção de vieses, quantificação de acertos ou erros e facilitação em análises de comparação, segundo Flake e Fried (2020).

Na avaliação dos modelos aqui apresentados serão observadas a acurácia, o F1-Score ponderado e o valor de AUC.

Acurácia

A acurácia mede a proporção de previsões corretas em relação ao total de amostras avaliadas:

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.4}$$

onde:

- TP = Verdadeiros Positivos
- TN = Verdadeiros Negativos
- FP = Falsos Positivos
- FN = Falsos Negativos

F1-score (ponderado)

O F1-score é a média harmônica entre precisão e revocação (recall), equilibrando a relação entre falsos positivos e falsos negativos. O valor ponderado leva em conta o suporte (quantidade de amostras) de cada classe.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precisão} \times \text{Revocação}}{\text{Precisão} + \text{Revocação}} \tag{2.5}$$

com:

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{2.6}$$

$$\text{Revocação} = \frac{TP}{TP + FN} \tag{2.7}$$

AUC (Área Sob a Curva ROC)

A métrica AUC (*Area Under the Curve*) representa a área sob a curva ROC, que relaciona a taxa de verdadeiros positivos (TPR) com a taxa de falsos positivos (FPR) em diferentes limiares de decisão. O valor ponderado leva em consideração a proporção de cada classe.

$$\text{AUC} = \int_0^1 TPR(FPR)\, d(FPR) \tag{2.8}$$

onde:

$$TPR = \frac{TP}{TP + FN} \tag{2.9}$$

$$FPR = \frac{FP}{FP + TN} \tag{2.10}$$

3 Revisão da Literatura

Para o avanço deste trabalho, uma etapa fundamental é a revisão da literatura sobre o tema abordado. Essa seção contém os detalhes e resultados encontrados.

3.1 Mapeamento Bibliográfico

Para fim de embasamento técnico da pesquisa, realizou-se um processo de busca e mapeamento bibliográfico com duas plataformas de conteúdo acadêmico, sendo elas: o Portal de Periódicos da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (2025) e o Google Acadêmico (GOOGLE, 2025). Nesse sentido, delimitou-se o uso de artigos, teses e dissertações que estivessem com acesso aberto de maneira gratuita, apresentasse relevância por citações, de acordo com o tema proposto e publicado num período recente.

A busca fez uso de descritores relacionadas à **discurso de ódio**, com ênfase em estratégias de **classificação** e **balanceamento de dados**, que são os fatores em destaques no desenvolvimento desse trabalho.

Em português, os descritores aplicados foram:

- Discurso de ódio classificação: com resultado de 7 artigos encontrados no Portal CAPES e aproximadamente 51.000 resultados no Google Acadêmico.
- Aprendizado de máquina discurso de ódio: retornou 2 artigos no Portal CAPES e cerca de 24.800 resultados no Google Acadêmico.

Para ampliar a busca utiliza-os descritores na língua inglesa, os sendo os termos utilizados listados a seguir:

- *Hate speech classification*: com 1.710 produções identificadas no Portal CAPES e cerca de outras 235.000 no Google Acadêmico.
- *Imbalanced classes hate speech*: gerando 12 resultados no Portal CAPES e aproximadamente 26.800 resultados no Google Acadêmico.

O uso de descritores em ambos os idiomas amplia o alcance de levantamento, garantindo uma visão das soluções propostas anteriormente na literatura internacional e nacional. Essa etapa foi essencial para identificar tendências e abordagens metodológicas já testadas.

3.2 Trabalhos relacionados

Rathpisey e Adji (2019) trazem o problema de desbalanceamento de classes nos casos de detecção de discurso de ódio, com uso de quatro tipos de reamostragem: *Random Oversampling (ROS)*, *SMOTE*, *ADASYN* e *Random Undersampling (RUS)*. Essas técnicas são aplicadas aos modelos de classificação tradicional: *Support Vector Machine*, *Naïve Bayes* e *Regressão Logística* — sendo o último destacado com *ROS*, com uma acurácia de 91% e um *F1-Score* de 0,95 superando os demais modelos. Os autores apontam que o uso de modelos mais simples associados com reamostragem podem ter um desempenho significativo, dando ênfase para a atenção em um pré-processamento adequado.

Putri et al. (2020) analisaram a detecção de discurso de ódio com dados extraídos de tweets na língua indonésia, comparando os algoritmos de classificação de *Naïve Bayes*, Perceptron Multicamadas, *AdaBoost*, Árvore de decisão e *Support Vector Machine*. O dataset composto por 4002 textos de tweets apresentavam conteúdos sensíveis relacionados à política, religião e etnia. Os autores apontam como resultado final o destaque ao algoritmo *Multinomial Naïve Bayes*, com métricas de *recall* de 93,2% e acurácia de 71,2% — mesmo sem o uso de *SMOTE*, por conta de sua característica de lidar com distribuição, o modelo se evidenciou o melhor no contexto de desequilíbrio de classes.

Asogwa et al. (2022) explorara o uso do *Support Vector Machine (SVM)* na tarefa de detecção de discurso de ódio, observando a interpretabilidade do modelo. No estudo, o *SVM* foi empregado para classificar conteúdos em duas categorias — ofensivo e não ofensivo — com base em um conjunto de dados extraído do repositório *UCI*. Fez-se uso de técnicas de extração de características e ajuste de hiperparâmetros com objetivo de otimização. Os resultados evidenciaram que o classificador *SVM* alcança uma acurácia de aproximadamente 99% sobre o conjunto de teste, sendo considerado uma solução viável para cenários de desbalanceamento.

No estudo de Das, Bhattacharyya e Sarkar (2023), destacou-se o uso do algoritmo de Floresta Aleatória como uma das principais abordagens de aprendizado de máquina para a detecção de discurso de ódio em conteúdos publicados no Twitter. Entre os modelos avaliados — incluindo *Regressão Logística*, *Gaussian Naïve Bayes*, *K-Nearest Neighbor*, Árvore de Decisão e *Support Vector Machine* — a Floresta Aleatória apresentou o melhor desempenho, alcançando uma acurácia de 98,2% na identificação de mensagens com teor de ódio. Os autores trazem uma observação sobre a capacidade do modelo capturar padrões sutis com uso da combinação de múltiplas árvores.

Abdurrahman, Irawan e Setianingsih (2020) revisaram a aplicação do *Light Gradient Boosting Machine (LightGBM)* com incorporação de técnicas como *Gradient-based One Side Sampling (GOSS)* e o *Exclusive Feature Bundling (EFB)* para a classificação de discurso de ódio no Twitter, para se obter uma alternativa em relação ao tradicional

Gradient Boosting Decision Tree (GBDT). Nos experimentos conduzidos com um conjunto de 1.000 tweets, o modelo *LightGBM* alcançou uma acurácia de 86,05% com uma taxa de aprendizado de 0,175, utilizando uma divisão de 70% para treino e 30% para teste. Os autores observam ainda sobre a importância de um ajuste adequado desses parâmetros é essencial para melhorar a detecção.

Na publicação de Charfi et al. (2024), destaca-se o uso do classificador Perceptron Multicamandas como uma das abordagens de aprendizado profundo para detecção de discurso de ódio em árabe, considerando a complexidade linguística por conta da diversidade de dialetos. O estudo faz uso do corpus *ADHAR*, que contém múltiplos dialetos como árabe padrão, egípcio, levantino, magrebino e do Golfo e categorias de discurso de ódio relacionado à nacionalidade, religião e etnia. Na apresentação dos resultados, o *MLP* obteve o melhor desempenho sendo o treino realizado a partir de características extraídas via *n-grams* de caracteres, com nível de acurácia e *F1-score* de 90% para a detecção de discurso de ódio e de 92% para a detecção de categorias.

Como exemplo de detecção de discurso de ódio, porém no contexto da língua portuguesa, tem como exemplo o trabalho de Silva e Serapião (2018) em que fez se a implementação de uma *CNN* com uso de *embeddings* de palavras extraídos com *Wang2Vec* e *Glove*. Nos resultados, os autores apresentaram melhorias a abordagens clássicas e modelos de referência, tendo como métricas o *F1-Score* e acurácia. Um ponto de destaque em construção de metodologia é o uso de funções otimizadoras como *RMSprop*, *Adagrad*, *Adadelta* e *Adam*, trazendo uma observação sobre a possibilidade de experimento com maior número de épocas e variações nos filtros de convolução e também indicam a necessidade de expansão de conjuntos de dados anotados para além de classificações binárias nas análises.

D'Sa, Illina e Fohr (2020) se aprofundaram no tema da classificação automática de discurso de ódio utilizando representações de palavras por *embeddings* e Redes Neurais Profundas (*RNNs*), fazendo o uso de arquiteturas como a *Bi-Directional Long Short Term Memory (Bi-LSTM)*. No estudo, o *Bi-LSTM* foi empregado como um dos classificadores para processar sequências de *embeddings* geradas por *fastText* e *BERT*, aplicando-se a um conjunto de dados de tweets rotulado em três classes (*hate*, *offensive* e *neither*). Os resultados finais destacaram que o *Bi-LSTM* apresentou desempenho levemente superior quando comparado a outras redes profundas, como *CNN* e *CRNN*, evidenciando sua capacidade de capturar dependências contextuais em sequências textuais.

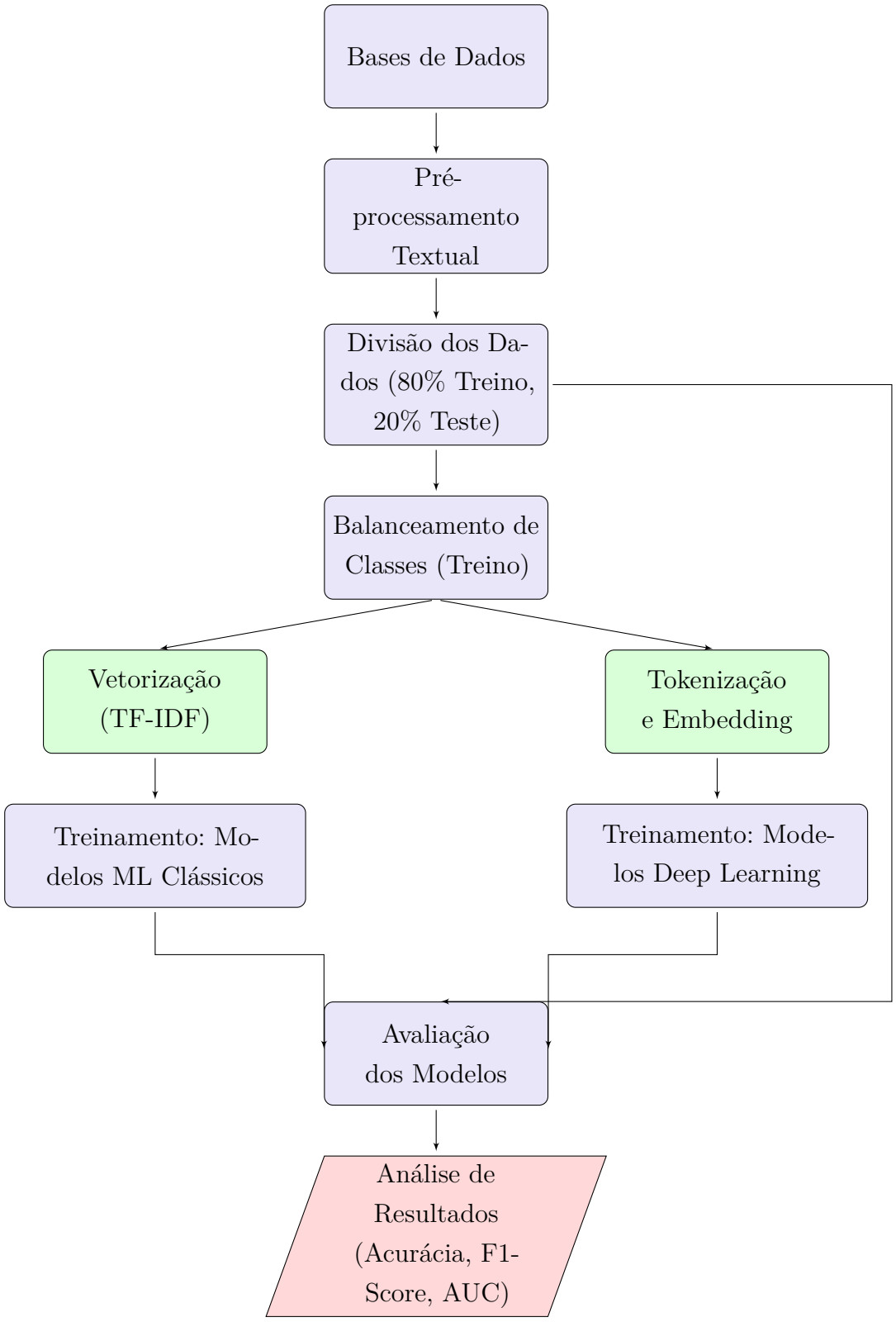
Um aspecto crítico discutido por Zhang, Hangya e Fraser (2024) é o desafio do desbalanceamento de classes na detecção de linguagem abusiva, uma vez que os dados com linguagem abusiva são classes minoritários nas bases de dados e isso afeta o desempenho dos modelos de maneira significativo. Os autores fizeram um estudo comparativo entre dez métodos conhecidos na literatura e trouxeram duas novas abordagens: *AbusiveLexiconAug*, que enriquece os dados com léxicos abusivos, e *ExternalDataAug*, que utiliza

conjuntos externos de dados abusivos. Nas considerações dos resultados, destacaram-se o uso de *oversampling* aleatório, *focal loss* e *AbusiveLexiconAug* para conjuntos de dados com características binárias ou *multi-classes* pequenas. Os autores sintetizam que não houve uma técnica universalmente superior às demais e reforçam o ajuste de critérios de hiperparâmetros na otimização.

4 Metodologia

O objetivo desse capítulo é informar sobre as bases de dados, algoritmos e métodos utilizados durante o trabalho. Para fornecer uma visão geral e estruturada das etapas executadas, a Figura 4 apresenta o fluxograma da metodologia adotada. As seções a seguir detalham cada uma dessas fases, desde a coleta dos dados até a avaliação final dos modelos.

Figura 4 – Fluxograma das etapas da metodologia.



Fonte: Elaborado pela autora.

4.1 Ambiente de execução

Para o desenvolvimento dos algoritmos, experimentação e testes dos modelos foi utilizado o ambiente do Google Colab[1] por meio de linguagem Python[2]. Essa escolha deu-se principalmente pelo fator de bibliotecas e pacotes auxiliares na construção da implementação, sendo os principais listados a seguir:

- **pandas**: Para manipulação e análise de dados, especialmente com DataFrames.[3]
- **numpy**: Para operações numéricas e trabalho com arrays.[4]
- **scikit-learn**: Para modelos de Machine Learning, divisão dos dados, pesos de classes e métricas dos modelos.[5]
- **nltk**: Para pré-processamento de linguagem natural, incluindo tokenização, stopwords e lematização.[6]
- **re**: Para operações com expressões regulares.[7]
- **tensorflow**: Para a construção e treinamento de modelos de deep learning (MLP, CNN, LSTM), incluindo a construção das diferentes camadas de redes neurais e tokenização e paddings de sequências.[8]
- **imbalanced-learn**: Para lidar com o desbalanceamento de classes, com uso de técnicas de oversampling e undersampling.[9]

O código-fonte da aplicação encontra-se disponível no repositório do GitHub, Oliveira (2025), acessível em: <https://github.com/ajuliasousa/TCC-2025-1>.

4.2 Base de dados

Foram utilizados três conjuntos de dados públicos, compostos por textos coletados de postagens feitas no Twitter e disponibilizados na plataforma Kaggle, com o objetivo de viabilizar projetos práticos com ênfase na detecção automática de discurso de ódio e linguagem ofensiva. A Tabela 1 apresenta um resumo dos principais conjuntos de dados utilizados, suas classes e quantidades de instâncias.

[1] <https://colab.google/>
[2] <https://www.python.org/>
[3] <https://pandas.pydata.org/>
[4] <https://numpy.org/>
[5] <https://scikit-learn.org/>
[6] <https://www.nltk.org/>
[7] <https://docs.python.org/3/library/re.html>
[8] <https://www.tensorflow.org/>
[9] <https://imbalanced-learn.org/>

Tabela 1 – Resumo dos conjuntos de dados utilizados

Dataset	**Total de Instâncias**	**Classes**	**Qtd**
Hate Speech and Offensive Language Dataset (Davidson et al., 2017)	24.783	Hate Speech	1.437
		Offensive Language	19.194
		Neither	4.152
Twitter Sentiment Analysis for Hate Speech Detection (Analytics Vidhya, 2022)	31.962	0 (Neutro)	29.725
		1 (Discurso de ódio)	2.237
Hate Speech Detection Curated Dataset (Nyantudre, 2020)	440.906	0 (Não odioso)	361.594
		1 (Discurso de ódio)	79.312

Fonte: Elaborado pela autora.

A primeira base de dados é denominada Hate Speech and Offensive Language Dataset (DAVIDSON et al., 2017), sendo feita por pesquisadores da Universidade de Cornell e da Universidade da Califórnia. Esta base reúne 24.783 tweets anotados manualmente e classificados em três categorias: hate speech (conteúdo que expressa ódio ou violência contra grupos específicos), Offensive Language (linguagem ofensiva) e Neither (neutro).

O segundo conjunto, Twitter Sentiment Analysis for Hate Speech Detection, disponibilizado pela Analytics Vidhya (VIDHYA, 2022), foca na detecção de discursos de ódio com conteúdo racista ou sexista. Cada tweet possui um rótulo binário: '1' para indicar discurso de ódio e '0' para conteúdo neutro. Para garantir privacidade, menções diretas a usuários foram substituídas por "@user".

Por fim, o terceiro dataset, denominado Hate Speech Detection Curated Dataset (NYANTUDRE, 2020), busca refletir características da linguagem contemporânea das redes sociais, incorporando emojis, hashtags, gírias e contrações comuns na comunicação online. As amostras são curtas, em inglês, e rotuladas em duas classes: '0' para conteúdo não odioso e '1' para discurso de ódio.

Assim, a combinação desses três conjuntos de dados possibilita uma análise abrangente, contemplando diferentes nuances linguísticas e níveis de complexidade na tarefa de detecção de discurso de ódio em redes sociais.

4.3 Pré-processamento

A etapa de pré-processamento textual pode ser dividida em basicamente dois passos: limpeza do texto e tokenização.

Na etapa de limpeza do texto, destacam-se as seguintes operações:

- remoção de URLs: limpeza de endereços de sites ou formatos de links
- remoção de menções: exclui nome de usuários após o uso de @, comum como forma de identificação nas redes sociais
- remoção de hashtags: elimina palavras que contenham o caractere especial # como marcação inicial
- filtro de caracteres não alfabéticos: são descartados numerais, pontuações e símbolos especiais
- conversão para minúsculas: todas as palavras são uniformizadas para fim de comparação
- remoção de espaços: exclusão de espaços extras ao longo das sentenças

Na etapa posterior, trata-se de um processamento avançado com as etapas descritas abaixo:

- tokenização: trata-se da fragmentação do texto em palavras individuais (os chamados tokens) para análise lexical em etapa posterior
- remoção de stopwords: exclui palavras frequentes como artigos, preposições e pronomes que não agregam valor semântico ao texto
- lematização: redução de cada palavra à sua forma canônica

Assim, as sentenças extraídas de postagem após esse processamento devem estar numa versão com menos ruídos, normalizadas e com tokens relevantes prontos para uso em vetorização para os modelos.

4.4 Algoritmos Utilizados

Inicialmente, têm-se foco nos modelos de classificação supervisionada que foram treinados para prever categorias de tweets, sendo a classe alvo aquela que identifica a presença de discurso de ódio.

A Regressão Logística foi utilizada como um modelo linear de base para classificação binária ou multiclasse, de acordo com as características do dataset em análise. Ela estima a probabilidade de um tweet pertencer a cada classe com base nas palavras mais relevantes, extraídas em etapa anterior com a técnica de TF-IDF. O modelo foi treinado com um número maior de iterações (equivalente a 1000) para garantir a convergência, dado o tamanho da matriz.

O modelo Naive Bayes Multinomial é adequado para tarefas de classificação de texto, por assumir que as características (ou seja, as palavras) ocorrem de forma independente. Deve-se ressaltar que por seu cálculo já considerar a distribuição não teve uso de métodos de balanceamento dentro da sua implementação nativa.

O Support Vector Machine (SVM) foi utilizado com kernel linear em sua configuração, uso comum para dados textuais de alta dimensionalidade, uma vez que se faz uso da matriz TF-IDF. Esse modelo busca encontrar hiperplanos que melhor separam as classes, sendo considerado robusto em tarefas de classificação.

A implementação do Random Forest foi treinada com 100 árvores de decisão, combinando os resultados dessas para obter mais estabilidade e a precisão da predição. Como um modelo de ensemble, ele lida bem com dados complexos e com menor sensibilidade a overfitting, se comparado a uma única árvore.

O LightGBM, um algoritmo de gradient boosting foi treinado diretamente sobre a matriz TF-IDF esparsa, com configuração para classificação binária ou multiclasse, de acordo com o dataset em questão.

Já a Rede Neural Multilayer Perceptron (MLP), construída com o Keras/TensorFlow, possui camadas densas com funções de ativação ReLU, camadas de dropout para reduzir overfitting e uma saída com softmax ou sigmoid, dependendo se o modelo for multiclasse ou binário, respectivamente. O modelo foi treinado por 20 épocas, com validação interna durante o processo, e sua performance foi avaliada também com base as métricas já mencionadas.

Para explorar a abordagem de aprendizado profundo , foi implementada uma rede neural convolucional (CNN) voltada para a classificação de textos. Como esse tipo de modelo trabalha fez-se uso de sequências de palavras, os tweets foram tokenizados e convertidos em sequências inteiras, com padding para garantir um comprimento fixo.

O modelo foi construído com uma camada de embedding (para mapear palavras em vetores densos), seguida por uma camada convolucional 1D que captura padrões locais no texto e uma camada de pooling que extrai as informações mais relevantes. Camadas densas e dropout foram adicionadas para refinar o aprendizado e reduzir overfitting. O modelo foi treinado por 10 épocas , sendo que a arquitetura visa capturar melhor a estrutura semântica dos textos nos conteúdos dos tweets.

O modelo baseado em Long Short-Term Memory(LSTM) foi desenvolvido como uma abordagem voltada ao reconhecimento de padrões sequenciais nos tweets. Utilizando a mesma base de dados preparada para a CNN, os textos foram previamente tokenizados e convertidos em sequências numéricas com padding, com comprimento uniforme.

A arquitetura do modelo inclui uma camada de embedding, que transforma cada palavra em um vetor denso, seguida por uma camada LSTM com 128 unidades, capaz de capturar dependências de longo prazo no texto. Foram adicionadas camadas densas e de dropout para auxiliar na generalização do modelo. A saída utiliza a função softmax para classificação multiclasse, quando classificação binária usa-se a função sigmoid.

O modelo foi treinado por 15 épocas e avaliado por meio de métricas comuns aos modelos anteriores, com objetivo de captar a natureza da ordem sequencial e contextual das mensagens de texto dos tweets.

4.5 Medidas de Avaliação

Cada modelo foi ajustado com os dados de treino e avaliado com os dados de teste usando três métricas principais: Acurácia, F1-score (ponderado) e AUC (Área Sob a Curva ROC, ponderada). Essas métricas são amplamente utilizadas para avaliar o desempenho de classificadores, especialmente em problemas de classificação balanceada ou desbalanceada.

Os resultados de cada métrica foram armazenados em um dicionário, possibilitando uma comparação estruturada e objetiva do desempenho entre os classificadores avaliados.

4.6 Configuração Experimental

A configuração experimental deste trabalho visa garantir a reprodutibilidade dos resultados e a comparação justa entre os diferentes modelos avaliados.

Para cada conjunto de dados, os tweets foram divididos em treino e teste utilizando a técnica de *holdout*, com 80% dos exemplos para treino e 20% para teste. Quando aplicável, foi realizada validação cruzada interna para ajuste de hiperparâmetros, isso no caso dos modelos baseados em redes neurais.

Os algoritmos que exigiam balanceamento de classes foram ajustados utilizando técnicas de *oversampling* ou atribuição de pesos de classe, com exceção do modelo Naive Bayes, que considera a distribuição das classes de forma nativa.

Os principais hiperparâmetros, como número de épocas de treinamento, batch size, número de árvores (no caso do Random Forest) e tamanho das camadas ocultas (em redes neurais) foram definidos com base em testes preliminares, visando otimizar o desempenho e reduzir o risco de *overfitting*.

5 Análise e discussão dos resultados

5.1 Resultados

Para avaliação dos modelos, foram testadas quatro condições: conjunto original desbalanceado, uso de *class weight*, oversampling e undersampling. As Tabelas apresentam os resultados obtidos em termos de Acurácia, F1-score Ponderado e AUC (OVR).

O primeiro dataset analisado é nomeado como *Hate Speech and Offensive Language Dataset* e reúne textos coletados do *Twitter*, sendo desenvolvido por pesquisadores da Universidade de Cornell e da Universidade da Califórnia, publicado originalmente no artigo "Automated Hate Speech Detection and the Problem of Offensive Language" Davidson et al. (2017).

Cada tweet da base foi anotado manualmente e classificado em uma das seguintes categorias:

- *Hate Speech* (Discurso de ódio): conteúdo que expressa ódio ou incita violência contra um grupo com base em atributos como raça, etnia, nacionalidade, gênero, orientação sexual, religião, entre outros.

- *Offensive Language* (Linguagem ofensiva): conteúdo que pode conter palavrões, xingamentos ou linguagem agressiva, mas que não necessariamente se configura como discurso de ódio.

- *Neither* (Nenhum dos dois): textos que não se enquadram em nenhuma das categorias acima, sendo neutros ou não ofensivos.

O dataset é composto por 24.783 tweets, porém apresenta característica de desbalanceamento entre as múltiplas classes. A distribuição das classes mostra que a maior parte dos tweets foi rotulada como linguagem ofensiva (77,4%), seguida por neutros (16,8%) e apenas uma pequena parte corresponde a discurso de ódio (5,8%).

Tendo esse contexto, parte-se para a implementação dos modelos apresentados na fundamentação teórica com o dataset original, em sequência com as técnicas de balanceamento para comparação de desempenho entre cada abordagem, conforme apresentado na Tabela 2.

Tabela 2 – Comparação dos Modelos no Dataset 1

Modelo	Métrica	Original	Class Weight	Oversampling	Undersampling
Regressão Logística	Acurácia	0.8923	0.8376	0.8396	0.7952
	F1-score Ponderado	0.8784	0.8538	0.8572	0.8222
	AUC (OVR)	0.9387	0.9313	0.9266	0.9177
Naive Bayes Multinomial	Acurácia	0.8322	0.8322	0.7906	0.7331
	F1-score Ponderado	0.7882	0.7882	0.8159	0.7810
	AUC (OVR)	0.8860	0.8860	0.8967	0.8886
SVM (Linear)	Acurácia	0.8983	0.8374	0.8363	0.7906
	F1-score Ponderado	0.8824	0.8572	0.8576	0.8212
	AUC (OVR)	0.9361	0.9409	0.9238	0.9259
Floresta Aleatória	Acurácia	0.8911	0.8907	0.8757	0.8158
	F1-score Ponderado	0.8769	0.8802	0.8787	0.8404
	AUC (OVR)	0.9392	0.9394	0.9341	0.9306
LightGBM	Acurácia	0.8963	0.8485	0.8723	0.7779
	F1-score Ponderado	0.8866	0.8646	0.8811	0.8142
	AUC (OVR)	0.9453	0.9386	0.9389	0.9212
MLP	Acurácia	0.8500	0.8300	0.8600	0.6500
	F1-score Ponderado	0.8464	0.8420	0.8456	0.7087
	AUC (OVR)	0.8944	0.8977	0.8815	0.8419
CNN	Acurácia	0.8700	0.8300	0.7300	0.7400
	F1-score Ponderado	0.8674	0.8455	0.7706	0.7895
	AUC (OVR)	0.9170	0.9078	0.8580	0.8975
LSTM	Acurácia	0.7700	0.0600	0.7700	0.7700
	F1-score Ponderado	0.6740	0.0065	0.6741	0.6741
	AUC (OVR)	0.5010	0.5994	0.5000	0.4985

De maneira geral, observa-se que os modelos tradicionais de classificação como Regressão Logística, *Linear SVM*, Floresta Aleatória e *LightGBM* apresentaram desempenho superior em termos de acurácia e *AUC*, com valores acima de 0,89 e 0,93, respectivamente, no cenário com uso do dataset original.

Para os modelos baseados em redes neurais (*MLP*, *CNN* e *LSTM*), os resultados variaram mais com o uso do balanceamento. Deve-se destacar que o *LSTM* apresentou um desempenho inconsistente, com acurácia de apenas 0,06 no cenário com balanceamento por meio do *class weight*, apontando uma dificuldade de generalização do modelo nesse caso.

Considerando as técnicas de balanceamento, observa-se que *class weight* e *oversampling* tendem a manter resultados próximos aos do cenário original, porém com relativo ganho no *F1-score* ponderado, indicador de melhora na classificação das classes minoritárias. Por outro lado, o uso do *undersampling* apresentou expressiva redução na acurácia e no *AUC* na maioria dos modelos, como se observa, por exemplo, na Regressão Logística (acurácia de 0,7952) e *Linear SVM* (0,7906).

Assim, no caso deste dataset, verifica-se que os modelos mais robustos em todos os cenários de teste foram a Floresta Aleatória e o *LightGBM*, mantendo *AUCs* próximas a

0,93 mesmo com as variações de balanceamento.

O segundo dataset foi disponibilizado por Vidhya (2022) publicamente e é intitulado como *Twitter Sentiment Analysis for Hate Speech Detection*. O objetivo principal é fornecer uma base para a detecção automática de discursos de ódio em publicações do *Twitter*, com foco específico em manifestações de cunho racista ou sexista.

Os dados são compostos por tweets com texto completo e um rótulo binário associado, sendo que o valor 1 indica a presença de discurso de ódio e o valor 0 representa textos livres desse tipo de conteúdo.

A etapa de estatística descritiva permitiu uma visão inicial da composição e distribuição dos dados utilizados neste projeto. O dataset de treino é composto por 31.962 tweets, sendo que a análise da distribuição mostra um forte desbalanceamento: apenas 7% dos tweets são classificados como contendo discurso de ódio, enquanto os demais 93% não apresentam esse tipo de conteúdo.

Já o dataset de teste contém 17.197 tweets e serve exclusivamente para avaliação, não incluindo os rótulos (*label*). Ambos os conjuntos não apresentam valores ausentes, e os identificadores (*id*) variam em um intervalo contínuo.

Em seguida, realiza-se a aplicação dos modelos com a variação de cenários de balanceamento. Os resultados obtidos estão sintetizados na Tabela 3, apresentada a seguir.

Tabela 3 – Comparação dos Modelos no Dataset 2

Modelo	**Métrica**	**Original**	**Class Weight**	**Oversampling**	**Undersampling**
Regressão Logística	Acurácia	0.9449	0.8911	0.8414	0.8392
	F1-score Ponderado	0.9295	0.9068	0.8723	0.8713
	AUC (OVR)	0.8940	0.8984	0.8772	0.8765
Naive Bayes Multinomial	Acurácia	0.9457	0.9457	0.8816	0.8070
	F1-score Ponderado	0.9300	0.9300	0.9003	0.8495
	AUC (OVR)	0.8659	0.8659	0.8926	0.8800
SVM (Linear)	Acurácia	0.9510	0.8839	0.8430	0.8201
	F1-score Ponderado	0.9409	0.9013	0.8729	0.8582
	AUC (OVR)	0.8591	0.8862	0.8542	0.8728
Floresta Aleatória	Acurácia	0.9550	0.9554	0.9162	0.8337
	F1-score Ponderado	0.9504	0.9506	0.9228	0.8676
	AUC (OVR)	0.8849	0.8904	0.8840	0.8808
LightGBM	Acurácia	0.9476	0.8781	0.8997	0.8078
	F1-score Ponderado	0.9378	0.8964	0.9105	0.8480
	AUC (OVR)	0.8625	0.8658	0.8556	0.8112
MLP	Acurácia	0.9500	0.9400	0.9500	0.8100
	F1-score Ponderado	0.9439	0.9394	0.9461	0.8539
	AUC (OVR)	0.8656	0.8606	0.8662	0.8695
CNN	Acurácia	*0.9400*	0.8900	0.7600	0.7800
	F1-score Ponderado	*0.9409*	0.9063	0.8189	0.8337
	AUC (OVR)	*0.8631*	0.8799	0.8038	0.8775
LSTM	Acurácia	0.9300	0.0700	0.0700	0.9300
	F1-score Ponderado	0.8962	0.0092	0.0092	0.8962
	AUC (OVR)	0.5004	0.5000	0.5001	0.5000

Na comparação entre modelos, a análise dos resultados mostra que os classificadores tradicionais como Regressão Logística, *Linear SVM*, Floresta Aleatória e *LightGBM* apresentaram desempenho consistente, com *accuracy* acima de 0,94 e *F1-score* ponderado superior a 0,92 no cenário original. Entre esses modelos, destaca-se a Floresta Aleatória, que obteve os melhores valores de acurácia e *F1-score*.

Em seguida, observa-se que os modelos baseados em redes neurais, como *MLP* e *CNN*, apresentaram resultados satisfatórios no cenário original do dataset (acurácia de 0,95 e 0,94, respectivamente), porém sofreram maior variação com o uso de *undersampling* e *oversampling*. O *LSTM* também se mostrou inconsistente e incapaz de prever a classe minoritária, apresentando valores de *AUC* equivalentes aos de um classificador aleatório.

Considerando as técnicas de balanceamento, nota-se que a aplicação de *class weight* e *oversampling* tende a reduzir ligeiramente as métricas de acurácia e *AUC*, porém se mantém estável ou até melhora o *F1-score* ponderado, indicando impacto positivo na classificação das classes minoritárias.

O terceiro dataset utilizado também encontra-se disponível na plataforma do *Kaggle* e é denominado *Hate Speech Detection Curated Dataset*, elaborado com o objetivo de

refletir as atuais tendências de linguagem presentes em plataformas de redes sociais, onde a propagação de discursos de ódio ocorre frequentemente por meio de conteúdo textual. O dataset reúne sentenças curtas em inglês anotadas em duas classes: 0 para conteúdo não odioso e 1 para conteúdo considerado discurso de ódio.

O dataset analisado é composto por 440.906 observações, cada uma representando uma sentença textual (coluna `Content`) anotada com um rótulo de classificação (`Label`) e uma representação numérica tokenizada (`Content_int`).

A coluna `Label` possui três valores distintos, com predominância da classe 0, que representa conteúdos não ofensivos, totalizando 361.594 ocorrências, cerca de 82% dos dados. As colunas `Content` e `Content_int` apresentam 417.561 valores únicos, indicando que a maior parte das mensagens e suas representações inteiras são distintas entre si, refletindo a diversidade textual do corpus.

Dado o grande volume de dados presentes no dataset original, com mais de 440 mil observações , optou-se pela criação de um subconjunto estratificado de 25.000 amostras para o desenvolvimento e experimentação dos modelos de *machine learning*. No subconjunto criado, foram selecionadas 20.503 instâncias da classe 0 (não odioso) e 4.497 instâncias da classe 1 (discurso de ódio), refletindo a distribuição original das classes.

Essa abordagem visa reduzir o custo computacional durante as etapas de pré-processamento, vetorização e treinamento, permitindo execução mais rápida e iterativa dos experimentos, sem comprometer significativamente a representatividade das classes. Os resultados de comparação são resumidos na Tabela 4, apresentada a seguir.

Tabela 4 – Comparação dos Modelos no Dataset 3

Modelo	Métrica	Original	Class Weight	Oversampling	Undersampling
Regressão Logística	Acurácia	0.8506	0.8054	0.7732	0.7802
	F1-score Ponderado	0.8281	0.8195	0.7921	0.8000
	AUC (OVR)	0.8609	0.8600	0.8276	0.8512
Naive Bayes Multinomial	Acurácia	0.8446	0.8446	0.7584	0.7356
	F1-score Ponderado	0.8175	0.8175	0.7828	0.7644
	AUC (OVR)	0.8494	0.8494	0.8549	0.8519
SVM (Linear)	Acurácia	0.8522	0.7954	0.7692	0.7742
	F1-score Ponderado	0.8372	0.8109	0.7886	0.7951
	AUC (OVR)	0.8339	0.8462	0.8167	0.8384
Floresta Aleatória	Acurácia	0.8440	0.8260	0.8108	0.7434
	F1-score Ponderado	0.8283	0.8184	0.8143	0.7692
	AUC (OVR)	0.8387	0.8357	0.8174	0.8281
LightGBM	Acurácia	0.8456	0.7928	0.7732	0.7652
	F1-score Ponderado	0.8269	0.8076	0.7913	0.7861
	AUC (OVR)	0.8414	0.8360	0.8298	0.8302
MLP	Acurácia	0.8400	0.8200	0.8400	0.7500
	F1-score Ponderado	0.8348	0.8195	0.8283	0.7708
	AUC (OVR)	0.8199	0.8155	0.8256	0.8147
CNN	Acurácia	0.8300	0.8100	0.7400	0.7400
	F1-score Ponderado	0.8206	0.8200	0.7625	0.7691
	AUC (OVR)	0.8076	0.8236	0.7328	0.8120
LSTM	Acurácia	0.8200	0.8000	0.6900	0.5600
	F1-score Ponderado	0.8203	0.8151	0.7218	0.6086
	AUC (OVR)	0.8144	0.8320	0.7150	0.7747

Os modelos tradicionais de Regressão Logística, *Linear SVM*, Floresta Aleatória e *LightGBM* mantiveram desempenho consistente em todos os cenários — sendo que os dois primeiros modelos se destacaram no cenário original, com acurácia acima de 0,85 e *F1-score* ponderado superior a 0,82.

Enquanto isso, os modelos *MLP* e *CNN*, baseados em redes neurais, apresentaram desempenho competitivo no dataset original, com acurácia entre 0,83 e 0,84, mas sofreram queda mais acentuada ao passarem por *undersampling*, com valores próximos de 0,74. O *LSTM* apresentou acurácia de aproximadamente 0,82 no cenário sem balanceamento, porém demonstrou maior variação na capacidade de generalização com o uso de *oversampling* e *undersampling*.

No que tange os métodos de balanceamento, observa-se que o uso de *class weight* e *oversampling* tende a manter o *F1-score* ponderado mais próximo do cenário original, ainda que haja leve redução na acurácia em alguns modelos. Já o *undersampling* provocou quedas mais acentuadas em praticamente todos os casos, apontando perda de informação relevante devido à exclusão de amostras.

6 Conclusão

Ao final desse trabalho e com os resultados obtidos com a aplicação de diversos modelos de *aprendizado de máquina*, *redes neurais* e técnicas de *balanceamento* apontaram para a robustez dos classificadores considerados tradicionais que foram mais consistentes em métricas, mesmo com o contexto de *desbalanceamento* de classes. Para os três conjuntos de dados, os modelos mantiveram *F1-Score* ponderados elevados e *AUCs* superiores a 0.93 no cenário original ou com uso de *class weight* e *oversampling*.

Já os modelos baseados em *redes neurais*, *MLP*, *CNN*, *LSTM* apresentaram sensibilidade maior à variação do tipo de *balanceamento* de classes. Sendo o último mais inconsistente na generalização, chegando a apresentar o desempenho de um classificador aleatório. Isso pode sugerir necessidade de ajustes adicionais na *arquitetura* ou no *pré-processamento* para esse tipo de modelo.

Dentre as técnicas de *balanceamento*, verificou-se que *class weight* e *oversampling* tendem a oferecer um equilíbrio entre custo computacional e melhoria na detecção de classes minoritárias, com impacto positivo no *F1-score* ponderado sem impactar muito a *acurácia* e o *AUC*. Já o *undersampling* teve o efeito de provocar redução de desempenho devido à perda de informações relevantes.

6.1 Limitações

Um fator limitante a se destacar foi o custo computacional em alguns experimentos, no caso de treinamento de *redes neurais profundas* e para a manipulação de grandes volumes de dados textuais. Assim, optou-se por *estratificação* de *subconjunto* seguindo a distribuição original, porém podendo ser um fator de alteração da representatividade dos resultados.

6.2 Trabalhos Futuros

Como perspectiva de aprimoramento, observa-se que há a possibilidade de realização de testes estatísticos de comparação *par a par* dos desempenhos dos modelos para validação da significância das diferenças percebidas. Pode-se fazer uso de teste como o de *McNemar* ou o teste de *Cochran's Q* para aplicar na verificação de variações do *F1-Score* entre cenários e modelos. Ainda para notação das flutuações de desempenho das classes minoritárias é possível o uso de *validação cruzada estratificada*.

Adicionalmente, para uma análise mais detalhada do processo de treinamento

dos modelos baseados em redes neurais, sugere-se a plotagem das curvas de aprendizado (treino, teste e validação) para diagnóstico visual do comportamento do modelo, como a ocorrência de *overfitting* ou *underfitting*.

Para trabalhos futuros, percebe-se possibilidades como: i) exploração de técnicas de *fine-tuning* dos modelos de *transformers* pré-treinados; ii) uso de *data augmentation* textual; iii) uso de *pipelines* de interpretabilidade de *Inteligência Artificial* para compreensão das classificações e até identificação de conteúdos sensíveis e por fim, iv) a investigação de modelos de detecção de anomalias, como os baseados em *One-Class Classification*, que podem oferecer uma abordagem alternativa e robusta para o problema, especialmente em cenários com classes de interesse raras e pouco definidas.

Referências

ABDURRAHMAN, M. H.; IRAWAN, B.; SETIANINGSIH, C. A review of light gradient boosting machine method for hate speech classification on twitter. In: *2020 2nd International Conference on Electrical, Control and Instrumentation Engineering (ICECIE)*. [S.l.]: IEEE, 2020. p. 1–6. Citado na página 34.

ASOGWA, D. C. et al. Hate speech classification using svm and naive bayes. *arXiv preprint arXiv:2204.07057*, 2022. Citado na página 34.

BREIMAN, L. Random forests. *Machine Learning*, v. 45, n. 1, p. 5–32, 2001. Citado na página 26.

CHARFI, A. et al. Hate speech detection with adhar: a multi-dialectal hate speech corpus in arabic. *Frontiers in Artificial Intelligence*, v. 7, p. 1391472, 2024. Citado na página 35.

Coordenação de Aperfeiçoamento de Pessoal de Nível Superior. *Portal de Periódicos da CAPES*. 2025. Disponível em: <https://www.periodicos.capes.gov.br/>. Citado na página 33.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine Learning*, v. 20, p. 273–297, 1995. Citado na página 26.

DAS, S.; BHATTACHARYYA, K.; SARKAR, S. Performance analysis of logistic regression, naive bayes, knn, decision tree, random forest and svm on hate speech detection from twitter. *International Research Journal of Innovations in Engineering and Technology*, v. 7, n. 3, p. 24, 2023. Citado na página 34.

DAVIDSON, T. et al. Automated hate speech detection and the problem of offensive language. In: *Proceedings of the 11th International AAAI Conference on Web and Social Media (ICWSM)*. [s.n.], 2017. p. 512–515. Disponível em: <https://ojs.aaai.org/index.php/ICWSM/article/view/14955>. Citado 2 vezes nas páginas 40 e 45.

DRUMMOND, C.; HOLTE, R. C. C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: *Workshop on Learning from Imbalanced Datasets II*. [S.l.: s.n.], 2003. Citado na página 31.

D'SA, A. G.; ILLINA, I.; FOHR, D. Classification of hate speech using deep neural networks. *Revue d'Information Scientifique & Technique*, v. 25, n. 01, 2020. Citado na página 35.

FLAKE, J. K.; FRIED, E. I. Measurement schmeasurement: Questionable measurement practices and how to avoid them. *Advances in Methods and Practices in Psychological Science*, v. 3, n. 4, p. 456–465, 2020. Citado na página 31.

FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, v. 29, n. 5, p. 1189–1232, 2001. Citado na página 27.

GIMPEL, K. et al. Part-of-speech tagging for twitter: Annotation, features, and experiments. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. [S.l.: s.n.], 2011. p. 42–47. Citado na página 24.

GOLDBERG, Y. *A Primer on Neural Network Models for Natural Language Processing*. [S.l.]: Morgan Claypool, 2016. Citado na página 29.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. Citado 2 vezes nas páginas 28 e 30.

GOOGLE. *Google Acadêmico*. 2025. Disponível em: <https://scholar.google.com.br/>. Citado na página 33.

HE, H.; GARCIA, E. A. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 21, n. 9, p. 1263–1284, 2009. Citado na página 31.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, v. 9, n. 8, p. 1735–1780, 1997. Citado 2 vezes nas páginas 29 e 30.

HOSMER, D. W.; LEMESHOW, S.; STURDIVANT, R. X. *Applied Logistic Regression*. 3rd. ed. [S.l.]: Wiley, 2013. Citado na página 26.

JOACHIMS, T. Text categorization with support vector machines: Learning with many relevant features. In: *European Conference on Machine Learning*. [S.l.]: Springer, 1998. p. 137–142. Citado na página 26.

JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing*. 3rd ed. draft. ed. [S.l.: s.n.], 2023. Disponível em: <https://web.stanford.edu/~jurafsky/slp3/>. Citado 2 vezes nas páginas 24 e 26.

KAUFMAN, L.; ROUSSEEUW, P. J. *Finding Groups in Data: An Introduction to Cluster Analysis*. [S.l.]: John Wiley & Sons, 2009. Citado na página 23.

KE, G. et al. Lightgbm: A highly efficient gradient boosting decision tree. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2017. v. 30, p. 3146–3154. Citado na página 27.

KIM, Y. Convolutional neural networks for sentence classification. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014. p. 1746–1751. Disponível em: <https://aclanthology.org/D14-1181/>. Citado na página 29.

LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, n. 11, p. 2278–2324, 1998. Citado na página 28.

MACAVANEY, S. et al. Hate speech detection: Challenges and solutions. In: *Proceedings of the 2019 AAAI conference on web and social media*. [S.l.: s.n.], 2019. p. 311–320. Citado na página 21.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. [S.l.]: Cambridge University Press, 2008. Citado 2 vezes nas páginas 24 e 27.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. Citado na página 25.

NYANTUDRE, A. *Hate Speech Detection Curated Dataset*. 2020. <https://www.kaggle.com/datasets/waalbannyantudre/hate-speech-detection-curated-dataset/data>. Citado na página 40.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. [S.l.: s.n.], 2014. p. 1532–1543. Citado na página 25.

PUTRI, T. T. A. et al. A comparison of classification algorithms for hate speech detection. In: *Iop conference series: Materials science and engineering*. [S.l.]: IOP Publishing, 2020. p. 032006. Citado na página 34.

RATHPISEY, H.; ADJI, T. B. Handling imbalance issue in hate speech classification using sampling-based methods. In: *2019 5th International Conference on Science in Information Technology (ICSITech)*. [S.l.]: IEEE, 2019. p. 193–198. Citado na página 34.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, Nature Publishing Group, v. 323, n. 6088, p. 533–536, 1986. Citado na página 30.

SCHMIDT, A.; WIEGAND, M. A survey on hate speech detection using natural language processing. *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, p. 1–10, 2017. Citado na página 21.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM Computing Surveys*, v. 34, n. 1, p. 1–47, 2002. Citado na página 27.

SILVA, A.; ROMAN, N. Hate speech detection in portuguese with naïve bayes, svm, mlp and logistic regression. In: *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*. Porto Alegre, RS, Brasil: SBC, 2020. p. 1–12. ISSN 2763-9061. Disponível em: <https://sol.sbc.org.br/index.php/eniac/article/view/12112>. Citado na página 21.

SILVA, S. C.; SERAPIãO, A. B. S. Detecção de discurso de ódio em português usando cnn combinada a vetores de palavras. In: *Symposium on Knowledge Discovery, Mining and Learning (KDMiLe)*. [S.l.]: SBC, 2018. p. 1–8. Citado na página 35.

SINOARA, R. A.; MARCACINI, R. M.; REZENDE, S. O. Mineração de textos e semântica: desafios, abordagens e aplicações. *Revista de Sistemas de Informação da FSMA*, v. 27, n. ja-ju 2021, p. 41–53, 2021. Citado na página 23.

STATISTA. *Hate Speech and Harassment Online - Statistics Facts*. 2023. Disponível em: <https://www.statista.com/topics/4235/hate-speech-online/>. Citado na página 21.

VIDHYA, A. *Twitter Sentiment Analysis for Hate Speech Detection*. 2022. <https://www.kaggle.com/datasets/arkhoshghalb/twitter-sentiment-analysis-hatred-speech/data>. Citado 2 vezes nas páginas 40 e 47.

ZHANG, Y.; HANGYA, V.; FRASER, A. A study of the class imbalance problem in abusive language detection. In: *Proceedings of the 8th Workshop on Online Abuse and Harms (WOAH 2024)*. [S.l.: s.n.], 2024. p. 38–51. Citado na página 35.